VŠB - Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

BAKALÁŘSKÁ PRÁCE

2014 Pavel Langer

VŠB - Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

Katedra informatiky

Desková hra - Mlýn

Chessboard Game

2014 Pavel Langer

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Pavel Langer**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Desková hra - Mlýn
Chessboard Game

Zásady pro vypracování:

Cílem bakalářské práce je navrhnout a implementovat aplikaci pro mobilní zařízení (tablet, smartphone). Aplikací bude desková hra Mlýn.

Práce bude obsahovat tyto části:
1. Popis deskové hry Mlýn (herní plocha a pravidla).
2. Objektový návrh aplikace. Návrh umělé inteligence počítačem řízeného hráče.
3. Návrh uživatelského rozhraní s přihlédnutím ke specifikům mobilního zařízení.
4. Implementace na vybrané mobilní platformě.

Seznam doporučené odborné literatury:

M. Zapletal: Velká kniha deskových her, Mladá fronta, Praha 1991, ISBN 80-204-0188-1

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **doc. Mgr. Jiří Dvorský, Ph.D.**

Datum zadání: 01.09.2013
Datum odevzdání: 07.05.2014

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně.

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne 7. května 2014

.....

Podpis

Poděkování

Rád bych poděkoval vedoucímu diplomové práce doc. Mgr. Jiřímu Dvorskému, Ph.D. za odbornou pomoc a konzultaci při vytváření této práce.

Abstrakt

Účelem této bakalářské práce je vytvořit mobilní aplikaci s přívětivým grafickým rozhraním (GUI) na platformu Android. Aplikace bude objektově navrhnuta s návrhem umělé inteligence počítačem řízeného hráče.

Klíčová slova

Android, umělá inteligence, GUI (grafické uživatelské rozhraní)

Abstract

The purpose of this bachelor thesis is to develop mobile application with a friendly graphical user interface (GUI) for Android platform. Application will have objective implementation with artificial inteligence computer playing player.

Key words

Android, artificial inteligence, GUI (graphical user interface)

Seznam použitých symbolů a zkratek

GUI	grafické uživatelské rozhraní (Graphical User Interface)
XML	rozšiřitelný značkovací jazyk (Extensible Markup Language)
UML	grafický jazyk pro vizualizaci (Unified Modeling Language)
TAMZ2	tvorba aplikaci pro mobilní zařízení
CPU	procesor (Central Processing Unit)

Seznam obrázků

Obrázek č.1 - Herní plocha

Obrázek č.2 - Ideální stav

Obrázek č.3 - Třídní diagram

Obrázek č.4 - Počátek aplikace

Obrázek č.5 - Rozehraná aplikace

Obsah

1. Úvod 2
2. Desková hra Mlýn 3
    A. Co jsou to deskové hry? 3
    B. Druhy deskových her 4
    C. Základní informace o hře Mlýn 5
    D. Pravidla hry 5
    E. Historie 6
    F. Strategie 7
3. Použité technologie 8
    A. Platforma Android 8
    B. Vývojové studio Eclipse 9
    C. Programovací jazyk Java 9
    D. Operační systém Mac OS X 10
4. Programovací část 11
    A. Popis Implementace 11
    B. Umělá inteligence 17
5. Závěr 22

Literatura 23

CD-ROM 24

1. Úvod

Hra Mlýn měla být vytvořena na libovolnou mobilní platformu (Windows Phone, Android, iOS), zvolil jsem si platformu Android kvůli její rozšířenosti a také díky zkušeností ze cvičení tvorba aplikací pro mobilní zařízení II. (TAMZ2).

Programovací část je provedeneá objektově do několika tříd. Hra obsahuje umělou inteligenci, při které počítač sám zhodnotí předchozí tah hráče a určí nejvhodnější tah pro sebe. Počítač první zjistí jestli může následující tah vytvořit takzvaný “mlýn" a tím odstranit nepřátelký žeton. Pokud nemůže vytvořit “mlýn”, tak se snaží zablokovat případný nepřátelský “mlýn” nebo vytvořit vlastní mlýn v následujích dvou tazích.

Na začátku této práce ve druhé kapitole jsou uvedeny pravidla deskové hry Mlýn a její historické prameny. V kapitole třetí jsou uvedeny veškeré použité technologie a základní informace o platformě Android pro mobilní zařízení. Ve čtvrté kapitole je vysvětlena programovací část aplikace, popis implementace a popis umělé inteligence. Poslední kapitolou je závěr, který shrnuje veškeré poznatky při vytváření téhle práce.

2. Desková hra Mlýn

A. Co jsou to deskové hry?

Kdyby jsme to měli definovat čistě logicky, tak bychom pravděpodobně řekli, že se jedná o druh stolních her, který se od jiných her liší tím, že herní děj probíhá na zvláštním plánku (hrací desce) a hraje se prostřednictvím různých hracích žetonů. Ale to jsme pouze řekli definici formální. Stanovit základní pojem —- hra je nesmírně obtížné. Někteří chápu hru jako organizovanou nebo kolektivní činost čistě rekreačního charakteru, ale někteří v tom můžou vidět mnohem více. Podle Holandského historika Johana Huitinga je pojem hra definována asi takto: [1]

"Hra je dobrovolná činnost, která je vykonávána uvnitř pevně stanovených časových a prostorových hranic, podle dobrovolně přijatých ale bezpodmínečně zavázaných pravidel, která má svůj cíl v sobě samé a je doprovázena pocitem napětí a radosti a vědomím 'jiného bytí', než je 'všední život'."

Jestliže tuhle definici aplikujeme na některou námi velmi dobře známou deskovou hru (například dámu nebo šachy), zjistíme, že je v tomto případě poměrně dost přesná.

Když bychom rozebrali řadu deskových her, tak bychom došli k možná překvapujícímu zjištění, že se od sebe hry zase tolik neliší a to nejen v podobě herního plánu a hracích žetonů, ale také samostatnou podstatou hry jako takové. Vedle her strategického typu jsou hry, které modelují zcela jinné životní situace. Při hře se každý z nás může stát dostihovým milionářem nebo nejlepším automobilovým závodníkem. Nad herní deskou v různě pozměněné mapě herního světa lze prožít Tolkienovou procházku středozemně, cestu kolem světa a to dokonce třeba za pouhých osmdesát minut.

Každá hra je ovšem spojována s jedním zvláštním fenoménem a tím je výhra a prohra. Při hře se člověk učí lépe snášet porážky, ale také třeba vítězství. Učíte se sebeovládání, strategickému nebo logickému myšlení a tak se hra sama stává vzorem

správného sociálního chování. Neboť i v běžném životě každý z nás musí dodržovat jistá “pravidla hry”.

B. Druhy deskových her

Každý z nás patrně zná celou řadu různých deskových her. Všechny tyhle hry jsme pochopitelně schopni zařadit do několika málo rozdílných skupin.

Strategické hry

Hlavním cílem strategických her je souboj dvou nebo více nepřátelských stran. Do téhle kategorie her patří například šachy, dáma, japonská hra Go, hra na vlka a ovečky, ale také právě hra deskový hra Mlýn, která je námětem téhle bakalářské práce. V každé z těchto her se nějakým způsobem zbavujeme nepřátelských žetonů z hracího pole.

Závodivé hry

Při závodivých hrých mají hráči za úkol dopravit svůj žeton po předem určené dráze co nejrychleji do cíle. Občas bývá závod i rozšířen o nějaký zbůsob boje. Asi nejpopulárnější zastánce závodivých her je slavná hra Člověče, nezlob se.

Poziční hry

U téhle kategorie her spolu hráči navzájem nezávodí ani nebojují, ale výhradně manévrují se svými žetony, tak aby vytvořili určitou sestavu nebo splnili zadaný úkol. Jako příklad téhle kategorie her bych uvedl třeba hru Piškvorky.

Pátrací hry

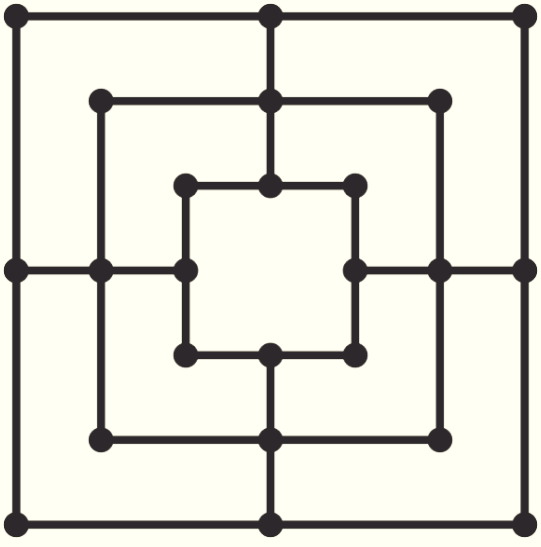
Poslední kategorií jsou tzv. pátrací hry, u kterých hráči řeší určitý problém. Také jsi ovšem mohou hrát na detektivy a tím pádem se pokouší vyřešit jistý problém.

C. Základní informace o hře Mlýn

Stolní hra Mlýn je strategická desková hra určená pouze pro dva hráče. Hra Mlýn pravděpodobně vznikla z jedné z variant her zvané Three Men's Morris (tento název pochází z angličtiny). Hra obsahuje obrovský počet platných tahů hry, které se odhadují až na neskutečných 10^{10} různých tahů, kdežto celkové množství možných kombinací hry je ještě mnohem větší a to až úctihodných 10^{50}. Na podzim roku 1993 přišel Ralph Gasser na herní systém, při kterém vždy dojde v tom nejhorším případě pouze k remíze. Ralph Gasser také navrhnul algoritmus nazývaný Bushy, tenhle algoritmus umělé inteligence je považován za vůbec nejlepšího hráče téhle hry na světě.

D. Pravidla hry

Celkem se hraje s osmnácti kameny, děvět bílých pro jednoho hráče a devět černých pro druhého hráče, tyhle kameny se poté pohybují po dvacetičtyřech průsečících. Pravidla hry jsou podobná jako u hry dáma; cílem hry je sebrat vašemu protihráči sedm kamenů z celkového počtu devítí kamenů nebo ho bloknout, tak aby neměl možnost dalšího tahu.



Obrázek č. 1 - Herní plocha

Rozmístění kamenů

Jako u většiny her i tahle začíná s prázdnou herní plochou. Oba dva hráči střídavě pokládají své žetony na hrací plochu. Jakmile hráč umístí tři kameny vedle sebe v jedné z propojených řad, tak hráč získává "mlýn" a musí odebrat jeden ze soupeřových žetonů. Jakmile je žeton odstraněn z herní plochy, tak se s nim již dále nesmí hrát. Jakmile se rozmístí všech devět žetonů obou hráčů a nedojde ke konci hry, tak se žetony jednotlivých hráčů začínají přesouvat po hrací ploše.

Přesouvání kamenů

Žetony se přemísťují po přímkách hrací plochy na vedlejší volnou pozici, pokud nemá hráč kam svůj žeton přemístit, hráč prohrává.

Stejně tak jako při rozmísťování žetonů i v této fázi hry platí, že pokud hráč přesune žeton tak, že vytvoří novou řadu tří žetonů nebo-li tzv. “mlýn”, musí odstranit nepřátelský žeton. Jakmile jeden z hráčů má pouze dva žetony, tak již nemůže sestavit “mlýn” a tím odstranit protivníkův žeton, tak pro něj hra končí.

E. Historie

Podle R. C. Bella, byla nejstarší herní plocha objevena vyrytá na střešních taškách chrámu Kurna v Egyptě, jejich vznik se odhaduje na období okolo roku 1400 př. n. l.

Berger věří, že hru mlýn z nějvětší pravděpodobností znali už staří Římané, neboť v jejich obydlích bylo nalezeno velké množství hracích desek, ale i tak není možné přesně stanovit jejich stáří z důvodu přístupnosti jejich obydlí i dalším generacím po tomto období. Je ovšem také možné, že se hra do Říma dostala skrze obchodní stezku.

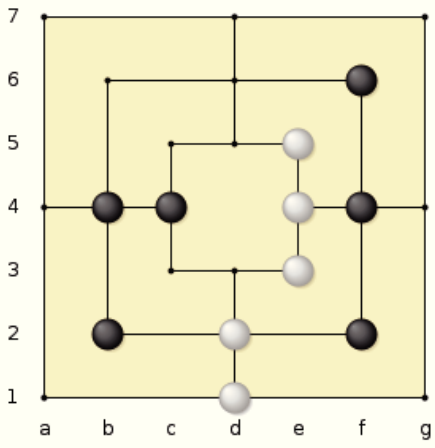
Hra si získala velkou popularitu také ve středověké Anglii. Právě v Anglii byly nalezeny herní desky vyřezané do sedadel v několika katedrálách. Na těhle hracích deskách byli místa pro jednotlivé kameny označené dírami namísto čar, odtud také hra dostala jméno "nine holes" (devět děr). Nicméně tímhle v Anglii nekončíme, tahle hra se také objevila v Shakespearově tvorbě ze 16. století a to konkrétně v jeho díle Sen noci svatojánské.

Historikové si ovšem nejsou vůbec jistí ohledně názvu téhle hry. Někteří tvrdí, že je jméno spjaté s tancem Morris, z kterého vznikl název Moorish, Daniel King nicméně tvrdí, že slovo “morris” nemá nic společného s názvem stejnojmenného starobylého anglického tance. Podle něho totiž název pochází z latinského slova “merellus", které označuje šachového pěšce nebo hrací kamen.

F. Strategie

Na začátku hry je vhodné si kameny rozmístit rovnoměrně po celé hrací ploše, není nutné se snažit ihned vytvořit "mlýn" a soustředit se tak pouze na jednu část herní plochy. Měli bychom se pokusit zabránit protihráči ve vytvoření mlýnu a připravit si dobrou výchozí pozici pro pozdější přesouvání žetonů.

Ideálním stavem, který později většinou vede k výhře je rozmístit si kameny tak, aby byl hráč schopen vytvářet mlýn na jedné, a následně v dalším tahu, na druhé pozici a tenhle tah opakovat, aniž by byl schopen jeho soupeř mu v tom tahu zabránit. Na obrázku č. 2 je vyobrazen tenhle ideální stav, dokonce i v případě, že by prvni táhnul hráč s černými žetony a snažil se zabránit protihráči ve vytvoření jeho "mlýnu", v dalším tahu si bílý hráč udělá "mlýn" a může černému hráči jeho žeton, se kterým se snaží zabránit vytvoření “mlýnu" bílému hráči odstranit.



Obrázek č.2 - Ideální stav
Obrázek převzat z wikipedie

3. Použité technologie

A. Platforma Android

Android [3] je velmi rozsáhlá open source platforma, která vznikla především pro různá mobilní zařízení. Obsahuje v sobě operační systém, který je založeny na Linuxovém jádru, middleware, uživatelské rozhraní a aplikace. Během vývoje systému se brala v úvahu jistá omezení, kterými disponují mobilní zařízení, tím se myslí například menší výkon nebo slabší výdrž baterie. Zároveň ovšem bylo Androidí jádro navrženo tak, aby zvládno běh na různých hardwarových platformách. Proto operační systém Android je možno využít bez ohledu na použitý chipset nebo třeba velikost obrazovky.

Samotný operační systém Android nedává k dispozici pouze operační systém s uživatelským rozhraním pro koncového uživatele, ale také kompletní řešení operačního systému pro mobilní operátory a výrobce zařízení a v neposlední řadě také pro samotné vývojáře aplikací poskytuje efektivní nástroje pro jejich vývoj a další rozvoj.

Mobilní aplikací vyvíjené pro operační systém Android probíhá v poupravené verzi známého a velmi populárního programovacího jazyka Java. Pro vykreslení uživatelského rozhraní (UI - User Interface) se používá XML soubor vytvořený v Android SDK (activity_main.xml). Aplikace pro Android se ovšem nemusejí vyvíjet pouze v programovacím jazyku Java, ale také za pomocí HTML5.

Jelikož nevlastním mobilní zařízení s Adroidem musel jsem si vystačit pouze s emulátorem, který se používá pro testování aplikací, díky němu není nutné vlastnit fyzické zařízení s operačním systémem Android. Emulátor je obsažen v Android SDK. Dá se tak velice dobře otestovat chování aplikace. Horší je to s emulováním hovorů, funkcí bluetooth nebo video a audio výstup, tyhle funkce jsem naštěstí při svém vývoji nepotřeboval testovat. U hry mlýn jsem si tedy vystačil pouze s emulátorem, na kterém jsem bezproblému testoval chování aplikace.

B. Vývojové studio Eclipse

Původně jsem zamýšlel, že bych to vyvíjel v oficiálním *Android Developer Studiu* od Googlu, ale po vyzkoušení jsem zjistil, že by to nebylo zrovna ideální, jelikož se to v té době *Android Developer Studiu* nacházelo pouze v alfa verzi.

Proto jsem nakonec zvolil vývojové prostředí Eclipse [5] s oficiálním Android SDK. Je to prostředí, se kterým jsme pracovali i na cvičeních ve předmětu TAMZ2 (tvorba aplikaci pro mobilní zařízení).

Využil jsem Eclipse, který zároveň obsahoval i plugin s Android SDK, který již obsahuje veškeré API knihovny a nástroje potřebné ke spuštění, vytváření, debugování a testování aplikací pro Android platformu.

C. Programovací jazyk Java

Základní vlastnosti

Vývoj Javy trvá již více než 20 let, konkrétně od roku 1991. Java [4] je momentálně jedním z nejpopulárnějších programovacích jazyků současnosti. Jejim soušasným vlastníkem je společnost Oracle, které do svého vlastnictví Javu získala začátkem roku 2010 kdy došlo k akvizaci firmy Sun Microsystems.

Java je velmi přenositelná, proto je často používán pro programy, které mají pracovat na různých systémech a zařízeních. Díky tomu, že se Java kód se kompiluje až při spuštění aplikace, tak je možné Java aplikace spouštět na nejrůznějších platformách a na operačních systémech včetně operačních systémů Windows, Linux, ale také Mac OS X.

Java má zjednodušenou syntaxi jazyka C a C++, odpadla ovšem většina konstrukcí, které způsobovaly programátorům problémy, na druhou stranu přibyla celá řada užitečných rozšíření. Díky tomu je určen pro psání spolehlivého softwaru. Používá tzv. silnou typovou kontrolu, což znamená, že každá proměnná musí mít definovaný svůj datový typ. Java také podporuje vícevlákonové aplikace. Je

dynamická, za chodu můžou být knihovny rozšiřovány o nové třídy nebo funkce a to jak z extérních zdrojů, tak vlastním programem.

Interpretace

Některé platformy nabízejí také přímou hardwarovou podporu pro jazyk Java. Nacházají se zde některé mikroprocesory, které dokáži spustit Javu hardwarově namísto softwarové emulace.

Syntaxe

Syntaxe Javy se do značné míry velice podobá syntaxi staršího programovacího jazyka C++ (rok vziku 1983). Ovšem Java narozdíl od programovacího jazyka C++, který kombinuje syntaxi pro generické, strukturované a objektově orientované programování je Java téměř výhradně pouze objektově orientovaný jazyk. Veškeré kódy psané v Jave jsou napsány uvnitř libovolné třídy a kromě primitivních datových typů je v Javě všechno objekt. Oproti C++ Java nepodporuje vícenásobonou dědičnost pro třídy ani přetěžování operátorů. Způsob jakým se komentují v Javě jednotlivé řádky kódu je prakticky totožný se zápisem v C++.

Nevýhody Javy

Oproti C++ jsou programy napsané v Javě pomalejší při startu aplikace. Je to dáno tím, že se musí první program přeložit a až poté se program spustí. Další nevýhodou je větší paměťová náročnost při běhů programů.

D. Operační systém Mac OS X

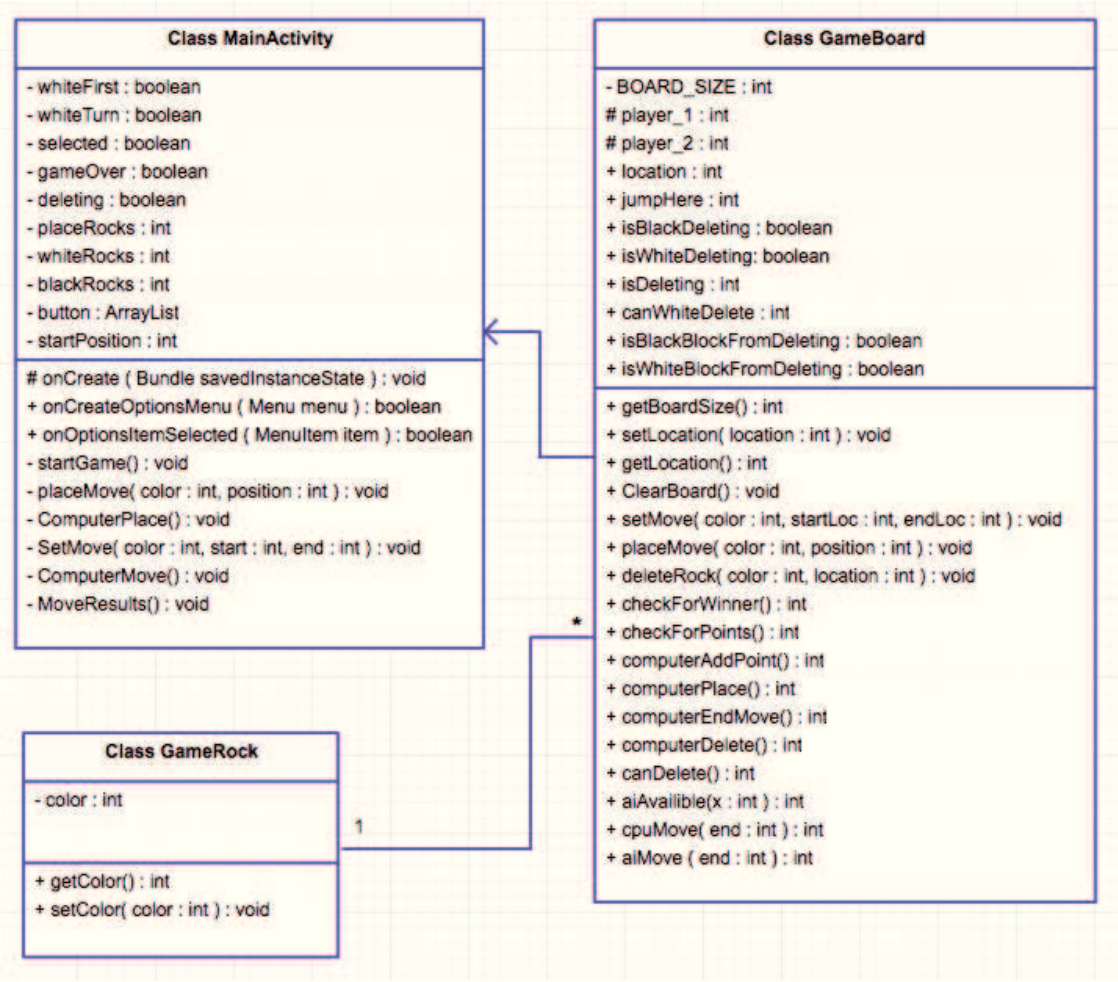
Aplikaci jsem vyvíjel na operačním systému Mac OS X Maverics, je to operační systém společnosti Apple vyvíjený od roku 1997. Oficiálně byl vydán v roce 2001 a jeho vývoj neustále pokračuje. Poslední verze byla vydána v říjnu minulého roku. Upravená verze Mac OS X nebo-li iOS využívají mobilní zařízení společnosti Apple (iPhone, iPad, iPod a také Apple TV).

4. Programovací část

A. Popis Implementace

Tato kapitola se stručně věnuje implementaci aplikace v programovacím jazyce Java.

UML - Třídní diagram



Obrázek č.3 - Třídní diagram

Třídy

Java je objektově-orientovaný programovací jazyk. Mezi hlavní rysy objektově-orientovaného programování patří dědičnost, zapozdření a polymorfismus. Zdrojový kód, který je napsaný v Javě po svém přeložení vytvoří třídu nebo několik tříd a všechny objekty, které se vytvářejí po spuštění přeloženého kódu jsou instancí tříd.

Třída MainActivity

V souboru *MainActivity.java* je hlavní třída *MainActivity*, která je rozšířena o třídu *Activity.* Obsahuje pět privátních proměných typu *TextView* pro zobrazení základních informací o stavu skóre jednotlivých hráčů a také informace o tom kdo je zrovna na řadě, případně textové pole s popisem; kdo hru vyhrál. Dále obsahuje privátní proměnné typu *button,* které jsou uloženy v poli, jednotlivé tlačítka vytvářejí hrací plochu. Ostatní proměnné jsou typu *boolean* nebo *int* a slouží k uchovávání počtu žetonů jednotlivých hráčů a zjištění, který hráč je momentálně na řadě, jestli ještě nedošlo ke konci hry, jestli hráč nezískal "mlýn" a tím pádem musí smazat jeden nepřátelský žeton a také proměnou *selected*, která je určena výhradně pro bílého hráče a jedná se o zjištění zda-li již hráč vybral nějaký žeton, se kterým se bude následně pohybovat po hrací ploše.

Při inicializaci se provede první metoda *onCreate().* Jejím úkolem je připravit layout pro aplikaci, zobrazit informační texty, vyčistit herní plochu od všech žetonů a nastavit do původních hodnot jednotlivé proměnné.

Poté se zavolá metoda *onCreateOptionsMenu()*, která vytvoří základní nabídku nebo-li tzv. menu. V menu jsou pouze dvě možnosti a to "Nová hra", která restartuje hru a začne se hrát znovu, jenom s tím rozdílem, že pokud v předchozí hře začínal bílý hráč jako první, tak teď začne hrát černý hráč jako první. Druhá možnost je "Ukončit hru", což jak název napovídá slouží pro ukončení aplikace.

Abstraktní třída ButtonClickListener

Abstraktní třída *ButtonClickListener* se nachází v hlavní třídě *MainActivity* a implementuje třídu *View.OnClickListener*. Tahle třída obsahuje metodu *ButtonClickListener*, která v sobě uchovává zakliknutou lokaci a dále obsahuje metodu *onClick*, která vykonovává určitou akci při kliku na dané tlačítko, které rozpozná pomocí už předem uložené lokace. Metoda *onClick* se bude vykonávat pouze v případě, že nedošlo ke konci hry, což rozpozná pomocí boolean proměné *gameOver*. Dále se potom zjišťuje jestli není nastavena boolean proměna *deleting* na true, pokud ano, tak musí hráč odstranit jeden nepřátelský žeton. Potom následuje kontrola jestli byli rozmístěny již všechny žetony na hrací plochu. Jakmile se rozmístí všechny

žetony, tak se kontroluje už jenom zda-li byl vybrán nějaký žeton bílého hráče, se kterým se bude následně hráč pohybovat po hrací ploše.

V případě booleanovské proměné *deleting* nastavené na true, bílý hráč odstraňuje černý žeton z hrací plochy. V první části se nastaví zakliklé tlačítko hrací plochy na obrázek bez žetonu, následně se tahle informace uloží do pole, proměna *deleting* se nastaví na false, aby nedošlo k dalšímu smazání žetonu v následujícm tahu. A opět se provede metoda *MoveResult*, která ukončí tah bílého hráče. V případě černého hráče se provádí odstranění žetonu bílého hráče v metodě *MoveResult*, kde se zavolá metoda *computerDelete*, která zhodnotí situaci a odstraní nejvhodnější nepřátelský žeton z hrací plochy.

Pokud bílý hráč rozmísťuje žetony na hrací plochu, tak po kliknutí na danou lokaci dojde ke změně obrázku na daném tlačítku na obrázek s bílým žetonem. Následně se zavolá metoda *placeMove*, která uloží na danou lokaci bílého hráče do pole, následně dojde k dekrementaci počtu žetonů bílého hráče potřebného k rozmístění a také celkového počtu žetonů, které je třeba rozmístit na hrací plochu, nakonec se přepíše text s počtem žetonů v informační části aplikace a zavolá se metoda *MoveResult*, která dále zhodnotí situaci.

Pokud rozmisťuje žetony černý hráč (CPU), tak se zavolá metoda *ComputerPlaceMove*, která zhodnotí situaci a rozhodne o nejlepším umístění černého hráče na hrací plochu a přepíše obrázek na vybraném tlačítku na obrázek s černým žetonem. Následně dojde k dekrementaci počtu žetonů černého hráče potřebného k rozmístění a celkového počtu žetonů, které je potřeba ještě rozmístit na hrací plochu, nakonec se přepíše text s počtem žetonů v informační části aplikace a zavolá se metoda *MoveResult*, která dále zhodnotí situaci.

Následující část metody *onClick* je určena pro pozdější část hry, kdy už jsou všechny žetony na hrací ploše rozmístěné. V téhle části bílý hráč označí žeton, který chce přesunout na novou pozici. Po výběru žetonu se nastaví proměna *selected* na true a zároveň se uloží aktuální pozice žetonu do proměné *startPosition* podle, které poté switch volá funkci *positionFunction* a ta rozhodne, zda-li se může hráč přemístit na novou pozici či nikoliv. Nakonec se zavolá metoda *MoveResult*, která opět překontroluje herní situaci a podle té vyhodnotí aktuální stav hry.

Třída GameBoard

V souboru *GameBoard.java* je hlavní třída *GameBoard*, která obsahuje metody a funkce spojené s hrací plochou. Obsahuje privátní statickou proměnou *BOARD_SIZE* datového typu integer určující počet míst na hrací ploše. Dále obsahuje třídu *GameRock* a dvě statické, privátní proměné integer s hodnotou 9, které slouží jako počet žetonů jednotlivých hráčů. Dále obsahuje integer pole *allowed* s hodnotami, které určují na kterých místech probíhá pohyb a rozmísťování žetonů. Dále tahle třída obsahuje několik základních metod typu get a set a konstruktor, který obsahuje metodu *ClearBoard.*

Tahle metado nastaví z pole *allowed* na hrací plochu místa, na kterých se může hra odehrávat. Pokud je na hrací ploše nastavena hodnota 3, tak se jedná o místo, kde může hra probíhat (kde může být položen černý nebo bílý žeton), v případě hodnoty 4 se jedná o zakázané místo na hrací ploše.

```
public void ClearBoard() {
	for(int i = 0; i < BOARD_SIZE; i++) {
		if(Arrays.asList(allowed).contains(i)) {
			gameBoard[i] = new GameRock(3);
		} else {
			gameBoard[i] = new Gam'Rock(4);
		}
	}
}
```

Příklad 4.1 ukázka kódu - Metoda ClearBoard

Dále obsahuje metodu *placeMove*, která příjmá dva argumenty a to barvu a konečnou lokaci. Tahle metoda slouží pro bílého hráče při rozmísťování žetonů. Když bílý hráč klikne na hrací ploše pro položení žetonu, zavolá se ze třídy *MainActivity* tahle metoda, která uloží zakliklou pozici na hrací plochu. Podobně funguje i metoda *setMove* jenom s tím rozdílem, že mimo barvy a konečné lokací příjmá ještě třetí argument a to počáteční lokaci žetonu. Při pohybu žetonů, hráč vybere žeton se kterým se chce pohybovat, takže jeho startovní lokace se změní na prázdné místo a jeho konečná lokace se změní na žeton bílé barvy.

```
public void setMove(String color, int startLoc, int endLoc) {
	gameBoard[startLoc].setColor(3);
	gameBoard[endLoc].setColor(color);
}
```

Příklad 4.2 ukázka kódu - Metoda setMove

V momentě kde některý z hráčů má již pouze dva žetony, tak hra končí. Pro tuhle kontrolu slouží metoda *checkForWinner*, která je volána ze třídy *MainActivity* v metodě *MoveResult* po každém ukončení tahu.

Dále tahle třída obsahuje metody a funkce, které slouží pro umělou inteligenci, počítačem řízenou simulaci černého žetonu. Obsahuje veřejnou proměnou integer *jumpHere*, která slouží pro uchovávání hodnoty při pohybu žetonu černého hráče po hrací ploše, tahle hodnota se poté volá v hlavní třídě *MainActivity*. Dále obsahuje několik metod, které slouží pro zhodnocení aktuální situace a vygenerování nejvhodnějšího umístění nebo přemístění žetonu černého hráče po hrací ploše.

Jednou z takových metod je metoda *computerPlace* sloužící v první části hry, kdy se rozmísťují žetony po hrací ploše, tahle metoda vykonává několik cyklů, které postupně dosazují černé a bílé žetony na hrací plochu a zhodnocují situaci. Tam kde zhodnotí situaci jako nejvhodnější pro umístění černého žetonu, tak tam také žeton ponechá. Jinak žeton zase z hrací plochy odstraní.

Přesouvání žetonů po hrací ploše v případě umělé inteligence, je trochu komplikovanější, potřebujeme vracet dvě hodnoty a to odkud se černý žeton přemisťuje a kde se přemístí. Proto jedna hodnota se vrací ve funkci *computerEndMove* do hlavní třídy *MainActivity* a druhá hodnota je uchovávana v globální proměnou *jumpHere*, o které jsem se již zmínil výše. Funkce *computerEndMove* v sobě uchovává pole *CPUPositions*, které se při každém volání aktualizuje a ukládá se do něj aktuální informace o pozici všech černých žetonů rozmístěných na hrací ploše. Následně se pracuje v cyklu právě s těmito hodnotami, které se postupně dosazují do metody *aiMove*, což je metoda, která nám sděluje jestli se černý žeton na aktuální pozici může přesunout na novou pozici a to jestli je nová pozice povolený pohyb a zároveň na dané pozici není obsažen jak černý, tak bílý žeton. Dále obsahuje metodu *canDelete*, jelikož v případě, že se vytvoří mlýn, tak se smí odstranit pouze jeden žeton, poté by měl hráč uvolnit mlýn a znovu jej vytvořit v

dalším kole, aby mohl odstranit další žeton ze hry. Proto metoda *canDelete* blokuje nebo odblokovává možnost odstraňování žetonů z hracího pole, projíždí se cyklem celá hrací plocha a kontroluje se zda-li již byl uvolněný některý z žetonů, který tvoří trojici žetonů na hrací ploše.

Třída GameRock

V souboru *GameRock.java* je hlavní třída *GameRock*, která uchovává barvu jako žetonů jako číselnou hodnotu. Kde číslo jedno reprezentuje bílou barvu žetonů, číslo dvě reprezentuje černou barvu žetonů, číslo tři potom reprezentuje prázdné místo a jako poslední číslo čtyři je reprezentováno jako zakázané pásmo. Tahle třída obsahuje pouze jednu privátní proměnou *color* s datovým typem int. Následně dvě metody *getColor* a *setColor* a konstruktorem obsahující barvu.

XML soubor activity_main

Soubor *activity_main.xml* je XML soubor, který obsahuje rozmístění jednotlivých komponent v aplikaci. Je v něm vygenerováno uživatelské rozhraní. Vybrány layouty, umístěny tlačítka a textová pole na definované pozice. Každé tlačítko, tak jako textové pole má své vlastní unikátní id, podle kterého je možné jednotlivé komponenty od sebe rozeznat a dále s nimi pracovat v hlavní třídě *MainActivity*.

V následujícím kódu je část z XML souboru, který vytváří *TableLayout* s jedním řádkem tabulky a uvnitř řádku je umístěno sedm tlačítek, které reprezentují hrací plochu. Na každém tlačítku je nastaveno unikátní id, jeho dynamické rozměry, typ tlačítka a jeho základní obrázek. Celkem je v tabulce umístěno sedm řádků a v každém řádků sedm tlačítek.

```
<TableLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <TableRow
            android:id="@+id/tableRow1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >
            <Button
                android:id="@+id/button1"
                style="?android:attr/buttonStyleSmall"
                android:layout_width="41dp"
                android:layout_height="41dp"
                android:background="@drawable/free_space" />
```

Příklad 4.3 ukázka XML - Grafické rozhraní

B. Umělá inteligence

Rozmístění žetonů

V první části hry se pokládají herní žetony na hrací plochu. První hru začíná vždy bílý hráč nebo-li člověk. Jakmile hráč položí svůj žeton dojde k dekrementaci počtu žetonů a zavolá se metoda MoveResult(). V metodě MoveResolt se kotroluje zda-li hráč hru již nevhrál, popřípadě jestli hráč nezíkal ve svém tahu takzvaný "mlýn" (tři žetony v řadě). Pokud ano, tak mu tah ještě nekončí a musí odstranit jeden protivníkův žeton z hrací plochy. Potom je na řadě černý hráč.

Černý hráč v první části hry zavolá funkci ComputerPlace(), která rozhoduje o tom, kde by měl černý hráč položit svůj žeton. Projíždí se cyklus a pokud se na hrací ploše nenachází bílý nebo černý žeton, tak dosadí černý žeton na dané místo na hrací ploše. Poté se zavolá metoda checkForPoints(), kde se kontroluje zda-li by černý hráč nezískal "mlýn", kdyby dosadil právě na tohle místo v hrací ploše. Pokud by na dané pozici získal mlýn, tak tam černý žeton položí v jinném případě se žeton smaže z hracího pole a pokračuje se v cyklu.

```
for(int i = 0; i < allowed.length; i++) {
	int x = allowed[i];
	if (gameBoard[x].getColor() != 1 && gameBoard[x].getColor() != 2)
	{
		gameBoard[x].setColor(2);
		if (checkForPoints() == 2)
		{
			placeMove(2, x);
			return x;
		}
		else {
			gameBoard[x].setColor(3);
		}
	}
}
```

Příklad 4.4 ukázka kódu funkce ComputerPlace()

Pokud se dojede na konec cyklu a černý hráč zjistí, že nemůže dosadit žeton na žádnou pozici, tak aby získal “mlýn”, tak podobný cyklus zkontroluje zda-li nemůže bloknout bílého hráče před získáním “mlýnu”. Při neúspěchu se černý hráč pokusí vytvořit alespoň dvojičku, takže dosadí černý žeton, tak aby v následujícím kole mohl dosadit žeton, aby získal “mlýn”.

Pokud je ovšem začátek hry a černý hráč neumístil ještě žádný žeton, tak rozmístění žetonu probíhá zcela náhodně. Černý hráč náhodně vybere pozici z pole. Pokud na dané pozici ještě není umístěný žádný žeton, tak jej zde umístí.

```
do {
       int idx = new Random().nextInt(allowed.length);
       move = (allowed[idx]);
} while(gameBoard[move].getColor() == 1 || gameBoard[move].getColor() == 2);

return move;
```

Příklad 4.5 ukázka kódu náhodného rozmístění

Po umístění žetonu černého hráče probíhá opět kontrola zda-li černý hráč nevyhrál nebo nezískal “mlýn” pomocí metody MoveResult().

Jak jsem už zmínil výše pro kontrolu zda-li nezískal černý nebo bílý hráč trojičku tzv. “mlýn” se používá funkce checkForPoints(), která využívá cyklus, který se inkremetuje o trojku a kontroluje žetony na hrací ploše.

```
public int checkForPoints() {
       for(int i = 0; i < checkPositions.length; i += 3) {
              int p1 = checkPositions[i];
              int p2 = checkPositions[i+1];
              int p3 = checkPositions[i+2];

              if(gameBoard[p1].getColor() == 2 &&
                 gameBoard[p2].getColor() == 2 &&
                 gameBoard[p3].getColor() == 2) {
                     isDeleting = i;
                     return 2;
              }
       }
       return 0;
}
```

Příklad 4.6 ukázka kódu pro kontrolu “mlýnu”

Přesouvání žetonů

Po rozmístění všech osmnácti žetonů (9 bílého hráče a 9 černého hráče) přichází na řadu přesouvání žetonů. Žetony se mohou přesouvat pouze mezi sousedními (spojenými), prázdnými místy.

Využívá se k tomu podobná funkce jako v předchozím rozmísťování jenom s tím rozdílem, že se musí vracet dvě hodnoty. Odkud se žeton přemístí a druhá kam se žeton přemístí. Vracet dvě hodnoty přes jednu funkci samozdřejmě není možné proto ukládám jednu hodnotu do globální proměnné a druhou vracím pomocí funkce. K zjištění jestli se může žeton přemístit na danou pozici z daného místa se využívá funkce aiMove, která vyhodnocuje přesouvání pomocí konečné hodnoty. Ve funkci se projede switch, který volá funkci a ta překontroluje zda-li je možné se na danou pozici přemístit (jestli je daná pozice prázdná nebo obsažená bílým nebo černým žetonem).

V první části funkce, se uloží do pole veškeré pozice černého hráče.

```
for (int i = 0; i < getBoard_Size(); i++)
{
	if(gameBoard[i].getColor() == 2) {
		CPUPositions[j] = i;
		j++;
	}
}
```

Příklad 4.7 ukázka kódu - uložení pozic CPU hráče do pole

Z uloženého pole pozic černého hráče se ve druhé části funkce kontoroluje jestli může černý hráč pohnout žetony, tak aby získal trojičku “mlýn”.

```
for(int i = 0; i < CPUPositions.length; i++) {
	setMove = aiMove(CPUPositions[i]);
```

Funkce aiMove vrátí, kde se může černý hráč pohnout se svým žetonem.

```
	jumpHere = CPUPositions[i];
	gameBoard[setMove].setColor(2);
	if (checkForPoints() == 2) {
		return setMove;
	}
	else {
		gameBoard[setMove].setColor(3);
	}
}
```

Příklad 4.8 ukázka kódu - přemísťování žetonu černého hráče

V další části kódu se kontroluje zda-li by černý hráč přemístěním ze staré pozice na novou získal “mlýn”. Pokud ano, vrátí se počáteční hodnota žetonu a konečná hodnota žetonu, kde má skočit do třídy MainActivity, kde se zajistí přepis aktuálních pozic do pole a následně přepíše jednotlivé obrázky na hrací ploše, tak aby odpovídali nově přemístěným žetonům.

Pokud ovšem vyhodnotí situaci a nemůže získat svým tahem trojičku, tak dochází k dalším cyklům, které se snaží vyhodnotit situaci a sestavit tak alespoň dvojičku a poté až v následujícím tahu trojičku, nebo se pokusí zablokovat budoucí trojičku nepřátelského / bílého hráče. První co černý hráč kontroluje před svým tahem, je zda-li může sestavit trojičku, pokud nemůže, tak kontroluje zda-li nemůže vytvořit v následujícím tahu bílý hráč trojičku až poté se snaží vytvořit dvojičku.

Odstranění žetonů

Jakmile ve svém tahu vytvoří černý hráč na hrací ploše trojici žetonů, tzv. “mlýn”, tak jeho tah ještě nekončí, ale bude muset znovu vyhodnotit situaci a odstranit jeden nepřátelský / bílý žeton z hrací plochy. Na to se zavolá funkce computerDelete, která hodnotu, kde se nachází nepřátelský / bílý žeton s nejvyšší prioritou. První se vyhodnocuje, zda-li nemá bílý hráč již vytvořenou trojičku žetonů, pokud nemá, tak odstraní žeton ze dvojice žetonů. Pokud ovšem nemá bílý hráč vytvořenou ani dvojičku žetonů, tak odstraní náhodný bílý žeton umístěný na hrací ploše.

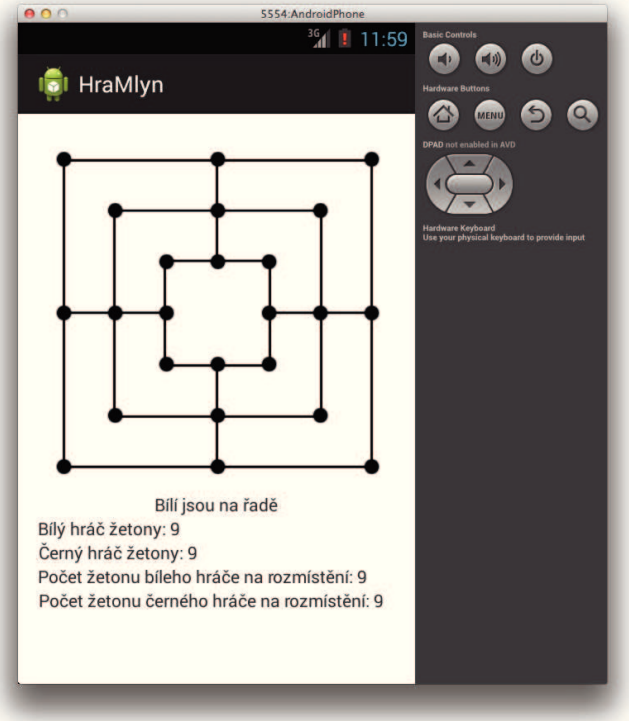
Následující ukázka kódu ukazuje, jak probíhá cyklus, ve kterém se odstraní jeden žeton bílého hráče, pokud má bílý hráč již vytvořenou trojičku.

```
for (int i = 0; i < getBoard_Size(); i++)
{
	if (gameBoard[i].getColor() == 1)
	{
		if (checkForPoints() == 1)
		{
			gameBoard[i].setColor(3);
			player_1 = player_1 - 1;
			return i;
		}
	}
}
```

Příklad 4.9 ukázka kódu - odstranění bílého žetonu z hrací plochy

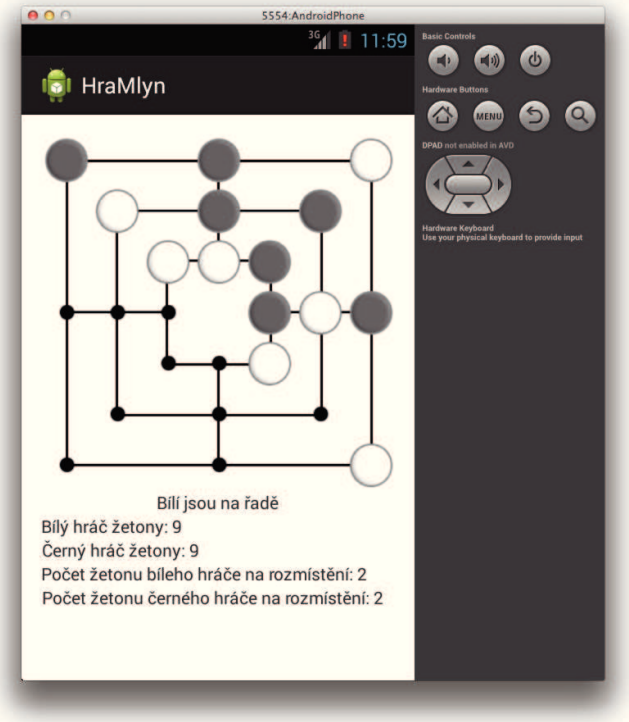
Ukázka uživatelského rozhraní aplikace

Při počátku hry ihned po spuštění aplikace je prázdná herní plocha a hra vypadá následnovně.



Obrázek č. 4 - Počátek aplikace

V průběhu hry při rozmísťování nebo přemísťování žetonů se herní pole začíná vyplňovat jednotlivými žetony a hrací plocha vypadá nějak podobně.



Obrázek č. 5 - Rozehraná aplikace

5. Závěr

Tahle bakalářská práce shrnuje autorovy poznatky programovacího jazyka Java pro Android při tvorbě herní aplikace získaného během bakalářského studia a samostudia jazyka Java.

Při zpravávání téhle bakalářské práce jsem si uvědomil jak je důležité na záčátku dobře zhodnotit a připravit si jak bude aplikace implemtována, jak se bude chovat a co od ní očekávám. V průběhu programování jsem totiž narazil na řadu problémů spjatých se špatným návrhem v počáteční fázi programování. Díky tomu jsem poté nebyl schopen rozdělit umělou inteligenci do samostatné třídy a musela být součásti již vytvořené třídy.

Důležitou součástí bylo vypracování umělé inteligence, při které bylo potřeba hlubšího se zamyšlení nad programovou problematikou. V následující straně je seznam veškeré literatury, která mě byla velmi nápomocná při psaní téhle bakalářské práce. Každý z těchto uvedených zdrojů sloužil jako pomoc při implementaci nebo jako zdroj odborných textů.

Literatura

[1] Miloš Zapletal: Velká Kniha Deskových Her,
Mladá Fronta, 1991

[2] Vávrů Jiří, Ujbányai Miroslav: Programujeme pro Android,
Grada Publishing 2013

[3] http://developer.android.com: Java Android Documentation

[4] http://www.oracle.com/technetwork/java/javase/documentation/index.html:
Oracle Jave dokumentace

[5] http://developer.android.com/tools/sdk/eclipse-adt.html: Eclipse

CD-ROM

Obsah CD-ROM:

- Tento text ve formátu PDF
- Zdrojové kódy hry Mlýn